4

Systems
Optimization
Laboratory

Interior-Point Methods for Linear Programming: A Challenge to the Simplex Method

by
Philip E. Gill, Walter Murray
and Michael A. Saunders

TECHNICAL REPORT SOL 88-14

July 1988

Department of Operations Research
Stanford University
Stanford, CA 94305

4

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305-4022

Interior-Point Methods for Linear Programming: A Challenge to the Simplex Method

by
Philip E. Gill, Walter Murray
and Michael A. Saunders

TECHNICAL REPORT SOL 88-14

July 1988

Research and reproduction of this report were partially supported by the U.S. Department of Energy Grant DE-FG03-87ER25030; National Science Foundation Grants CCR-8413211 and ECS-8715153; Office of Naval Research Contract N00014-87-K-0142, and Air Force Office of Scientific Research Grant AFOSR 87-0196.

Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do NOT necessarily reflect the views of the above sponsors.

INTERIOR-POINT METHODS FOR LINEAR PROGRAMMING: A CHALLENGE TO THE SIMPLEX METHOD

Philip E. GILL, Walter MURRAY and Michael A. SAUNDERS

Systems Optimization Laboratory, Department of Operations Research
Stanford University, Stanford, California 94305-4022, USA

Technical Report SOL 88-14*
July 1988

Keynote Address, 24th Annual Conference
Operational Research Society of New Zealand
Auckland, New Zealand, August 18–19, 1988

Abstract

The world of mathematical programming has seen a remarkable surge of activity following publication of Karmarkar's projective algcrithm in May 1984. A review of the ensuing three years has already appeared. ~~(Monma [31])~~ One year later, ~~we~~ review some of the main methods and surrounding events, and focus on references that contain computational results.

Keywords: Linear programming, interior-point methods, barrier-function methods, Newton's method, Karmarkar's projective method,

1. Introduction

It is still only four years since Narendra Karmarkar of AT&T Bell Laboratories presented the mathematical features of an apparently new method for solving linear programming (LP) problems [23]. The problem was assumed to be in the form

$$\underset{x}{\text{minimize}} \quad c^T x \quad \text{subject to} \quad Ax = 0, \quad e^T x = 1, \quad x \geq 0, \qquad (1)$$

where A is $m \times n$, $m \leq n$ and e is a vector of 1's. It was also assumed that $x = e$ is a feasible solution, and that the optimal objective value is zero. These and other restrictions have since been lifted (though not without practical difficulty).

*The material contained in this report is based upon research supported by the Air Force Office of Scientific Research Grant 87-01962; the U.S. Department of Energy Grant DE-FG03-87ER25030; National Science Foundation Grants CCR-8413211 and ECS-8715153; and the Office of Naval Research Contract N00014-87-K-0142.

A vital part of Karmarkar's analysis involved showing that the so-called *potential function* $c^Tx/\prod x_j$ could be reduced by a fixed amount every iteration. It followed that even in the worst case, the number of iterations required to reach a solution would be proportional n. The work per iteration was shown to be of order $n^{2.5}$. Hence, the total time required was bounded by a polynomial in n.

Broadly speaking, the bound on computation time for the projective method is of order $n^{3.5}L$ (where L is a measure of the storage needed for the input data). In more recent methods the bound has been reduced to $O(n^3L)$ [34,37], whereas for the classical simplex method [4] the total time on certain contrived problems is of order $n^2 2^n$.

1.1. A practical algorithm?

To operations researchers and others, it seemed immediately evident that the amount of work per iteration for the new method would be *much greater* than for the simplex method, at least for "normal" LP problems. The crucial part of the computation involves a Cholesky factorization $AX^2A^T = R^TR$ (with X diagonal, R triangular), and this is normally much more expensive in time and storage than the sparse basis factorization $B = LU$ required by the simplex method.

Nevertheless, it seemed that the reduced number of iterations might sometimes compensate, and for *very specially structured problems* it was reasonable to suppose that the Cholesky factors could be obtained quickly enough to give an efficient algorithm.

1.2. Not just practical

Controversy soon enveloped the new algorithm. In a series of conference appearances that are continuing to this day, Karmarkar has reported results that are *uniformly superior* to those obtained by the simplex method. The speed-up factors claimed vary from 4 up to 84, 104, 190, 720 and even higher [24].

The factor of 4 was obtained using *part* of a set of 13 publicly available problems, supplied on request by Stanford University in July 1984. This test set was later dismissed as "not representative of modern real-world problems in size or complexity", even though several problems were omitted in the results reported, including the problem that was by far the largest! (The PILOT model had about 1500 rows, 3700 columns and 43000 nonzeros—certainly only medium-scale by conventional standards, yet "large" in the sense that a cold-start solution with the simplex method takes over 20 hours on a DEC VAXstation II.[1])

Although much larger problems are indeed of interest, credibility at the time depended strongly on results for familiar models. Omission *without comment* of the known most-difficult test cases inevitably generated scepticism.

Further controversy arose from the fact that the more spectacular results were obtained from problems that were proprietary, like the implementation itself. Neither the problems nor the program could be seen by the outside world, and apart

[1] This is a typical run-time for MINOS 5.3 (May 1988), a portable Fortran code. A commercial Mathematical Programming System would take about 30 minutes on an IBM 3090.

from problem dimensions, the only results given were iteration counts and CPU time. No plot of the individual iterations, no stopping criteria, no details about the precision obtained or the *reliability* of the code.

1.3. A flurry of effort

In spite of the unsatisfactory circumstances, much feverish activity has been generated within the scientific community, and numerous researchers have launched into projects they did not expect to be pursuing.

This is not entirely due to a windfall of government funding! There is also the intrigue and the challenge.[2]

1.4. An objective view

The connections with more traditional areas of nonlinear programming and numerical linear algebra, along with much analysis of path-following methods (e.g., [29,30,35]) have cast sufficient light on the scene for us to believe that good performance is indeed possible on a significant proportion of real-world problems.

In terms of robustness the verdict is still out, since present implementations (within our experience) are highly sensitive to slight changes in strategy. However, it took four decades to make the simplex method (virtually) 100% reliable. Much has been learned in that time about linear and nonlinear optimization. While the methods inspired by Karmarkar are certainly difficult to make "bullet-proof", there is every hope that useful and acceptably reliable implementations will be developed within the next few years.

In the following sections we review the main mathematical variants that have been proposed so far, giving known computational results where possible.

2. Primal and Dual LPs

The special form (1) allowed certain geometrical arguments, and many research papers have been written around it. However, operations researchers have long been generating linear programs in the more practical form

$$\text{Primal:} \qquad \underset{x}{\text{minimize}} \;\; c^T x \;\; \text{subject to} \;\; Ax = b, \quad x \ge 0, \qquad (2)$$

with no assumptions about the optimal objective value. Variants of the Karmarkar approach began to appear for solving (2) directly (without converting it to the special form); e.g., [12,18]. It was soon found that dual variables could be obtained as by-products. Algorithms for the dual problem

$$\text{Dual:} \qquad \underset{\pi}{\text{maximize}} \;\; b^T \pi \;\; \text{subject to} \;\; A^T \pi \le c, \qquad (3)$$

[2] The simplex method has been extremely successful so far, and computers are growing ever faster, yet it is never enough. Factors of 100 or more are tantalizing and compelling, even in the face of scepticism.

also began to appear. We discuss the deveiopments for both problems in roughly chronological order. The most general case with upper and lower bounds on x is covered in Section 6.

3. Newton's Method and Barrier Functions

Given a vector x, let X be a diagonal matrix whose j-th diagonal is x_j. It will be assumed that $x_j > 0$ for $j = 1$ to n. Similarly for quantities z and Z.

The *barrier function* approach deals with inequality constraints such as $x \geq 0$ by adding a judicious function to the true objective. A characteristic is that along any direction of descent from an interior point, there exists a strictly interior minimizer. Usually the transformed objective function is infinite along the boundary of the feasible region. Common barrier functions are the logarithmic and reciprocal functions.

3.1. A primal barrier method

For the primal LP we consider subproblems of the form

$$\underset{x}{\text{minimize}} \quad F_\mu(x) = c^T x - \mu \sum_{j=1}^{n} \ln x_j \quad \text{subject to} \quad Ax = b, \tag{4}$$

where μ ($\mu > 0$) is a specified parameter that will be set to decreasing values. As $\mu \to 0$, the solution of (4) approaches that of (2).

One of the first computational studies outside AT&T was undertaken by Gill *et al.* [18], who recognized the connection between Karmarkar's method and Newton's method applied to the barrier subproblems. Newton's method obtains a *search direction p* by solving a system of the form

$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} -p \\ \pi \end{pmatrix} = \begin{pmatrix} g \\ 0 \end{pmatrix},$$

where g and H are the first and second derivatives of $F_\mu(x)$.

Given a feasible interior point x, an estimate of the dual variables π and the corresponding "reduced costs" $z = c - A^T\pi$, the main steps prove to be as follows:

1. Define $r = z - \mu X^{-1}e$.
2. Solve the system
$$AX^2A^Tq = AX^2r, \quad \text{or equivalently,} \quad \min \|X(r - A^Tq)\|_2. \tag{5}$$
3. Update $\pi \leftarrow \pi + q$.
4. Define $z = c - A^T\pi$, $r = z - \mu X^{-1}e$ and $p = -(1/\mu)X^2r$.
5. Find a steplength α at which the barrier function $F_\mu(x + \alpha p)$ is suitably less than $F_\mu(x)$.

6. Update $x \leftarrow x + \alpha p$.

The algebra is evidently not much more complicated than in the simplex method. However, Step 2 is crucial. The matrix AX^2A^T is *large*, it could be quite *dense* compared to A, and in general it is very nearly *singular*. A combination of direct and iterative methods may be applied, but the system must be solved *accurately* in order to retain the condition $Ax = b$.

If the barrier parameter μ is held constant, the vector r will eventually become small as Steps 1–6 are repeated. The iterations then start again with a smaller μ. In practice, we typically give μ the values 0.1, 0.01, ..., 10^{-6} (say) over a total of 20–50 iterations.

3.2. Relationship to Karmarkar's projective method

The best choice of μ remains open to question. In [18] it was shown that if the Newton barrier method is applied to the special LP problem (1), and if μ and α are chosen in a certain (easily computed) way, then the sequence of points x are identical to those generated by Karmarkar's original method. In other words, the "new" method for linear programming may be viewed as a special case of Newton's method for linearly constrained optimization.

This is not to minimize Karmarkar's contribution. For example, the "special case" argument was deficient without a proof that the special choice of μ would be positive.

One practical contribution has been Karmarkar's unrelenting advocacy of the approach, *even though it requires solution of systems involving AX^2A^T*. Years earlier, prior to modern sparse-matrix techniques (e.g., [14]), researchers had rejected similar approaches as being unacceptably slow on large problems.

In addition, Karmarkar's particular choice of μ and α enabled him to obtain a polynomial bound on the iterations and work required. (No such bound was previously known for barrier methods.) Recently, Gonzaga [22] has shown that when Karmarkar's method is generalized to treat problem (2) it again corresponds to the barrier method with a certain *positive* choice of μ. Hence, the projective method and the primal barrier method are now known to be polynomial-time algorithms even on the more useful formulation (2).

Note that the Newton-barrier approach has practical and theoretical benefits. In contrast to Karmarkar's original algorithm, the extension to other forms of LP and to nonlinear programs is trivial. The Newton-barrier approach is also the basis of the analysis in [34,37].

3.3. Results

In [18], results were given for nine of the problems sent to Karmarkar (see Section 1.2); the remainder had upper bounds and could not be run. In eight of the nine cases, the primal barrier method was found to be comparable in speed to the simplex method, and on two additional (larger) models a speed-up factor of 3 was obtained.[3,4]

Unfortunately, a further series of highly degenerate models showed an unfavorable trend with increasing problem size. Nevertheless, the conclusions in [18] struck an optimistic note for interior-point methods, particularly for problems with special structure.

More recently, a new primal barrier implementation has been developed at Stanford [16]. Of particular interest is the solution time for PILOT: about 9 hours on a VAXstation II. This is a speed-up of 2.3 on a real-world model that is unquestionably non-trivial for the simplex method. The periodic structure revealed in [26] may be a contributing factor, but in any event, this represents a bright note for the interior-point approach within the scope of current repeatable computational results.

3.4. A dual barrier method

The barrier subproblem corresponding to the dual LP (3) is

$$\underset{\pi}{\text{maximize}} \quad b^T\pi + \mu \sum_{j=1}^{n} \ln(c_j - a_j^T\pi),$$

which is a purely *unconstrained* problem. For certain x and X [19], Newton's method leads to the equations

$$AX^2A^Tq = \mu(Ax - b), \qquad \pi \leftarrow \pi + \alpha q, \tag{6}$$

and we can capitalize on the fact that the system for q *need not be solved exactly* [5]. This opens to door to obtaining "cheap" approximate factors of the matrix AX^2A^T, for use as preconditioners with the conjugate-gradient method.

3.5. Results

For reasons described by Gill *et al.* [19], a sparse factorization $XA^T = LU$ should provide a useful preconditioner. An experimental implementation has so far proved to be less efficient than hoped. However, we anticipate that the advantages of approximate factors will ultimately come to the fore (see Section 7.3).

4. Affine Scaling

Independently of the barrier-function development described above, groups were at work on algorithms for the primal problem (2) and the dual problem (3). These were the so-called *affine variants of Karmarkar's method*, which seemed at least to be simpler than the projective method.

[3] The results reported here and in later sections were obtained using "cold starts". The ability to utilize "good" starting information remains virtually unique to the simplex method. The entropy-based method of Eriksson and the shifted barrier method also have this advantage; see Section 8.

[4] Comparisons with the simplex method were made using various versions of MINOS [36] without scaling or partial pricing. For future experiments we would in general recommend specifying SCALE OPTION 2 and PARTIAL PRICE 10; see Lustig [26].

4.1. A primal affine method

At Bell Labs, Vanderbei *et al.* [40] developed an algorithm that proved to be the limiting case of the primal barrier method as $\mu \rightarrow 0$ [18]. It was subsequently found that the method was first proposed by Dikin in 1967 [6].

Search directions are generated by the same system (5). Any hint of quadratic convergence (associated with Newton's method) is lost by taking $\mu = 0$, and to date there has been no proof of polynomial-time complexity. Nevertheless, respectable computational results were obtained from a production code developed at Bell Labs (Chen [3]).

We believe that a primal affine algorithm is embodied within a combined software/hardware system that is currently being marketed by AT&T: the KORBX$^{\text{TM}}$ Linear Programming System, which includes an Alliant multi-processor computer.

4.2. Dual affine methods

Under the guidance of Karmarkar, researchers at Berkeley [2] developed an analogous dual algorithm that again proved to be the limiting case of the (dual) barrier method as $\mu \rightarrow 0$. Several advanced implementation techniques were employed [1], and promising results were obtained (see next section).

A similar implementation was developed in 1986 at Bell Communications Research by Monma and Morton [32].

4.3. Results

There is now a collection of over 50 test problems available publicly via *netlib* (see Gay [10], Lustig [26]). About 30 of these problems were used to test the dual affine implementations just mentioned. Both codes achieved average speed-ups of 3 relative to MINOS (versions 4.0 and 5.0 repectively).

As before, some of the more interesting problems were not tested because the implementations could not handle upper bounds. We remark that the algorithmic parameters needed for satisfactory performance on *all* of the problems would probably give poorer average performance on the 30-problem subset.

Until recently it was believed that a large speed advantage was arising in the Berkeley implementation [2] from the use of in-line code (and large amounts of memory) during the Cholesky factorization [1]. However, Gay [13] has now shown that such an advantage need not exist.

5. Primal-Dual Methods

Barrier functions have recently been taken up by several authors in a primal-dual context (e.g., see Monteiro and Adler [33], Lustig [27]). Briefly, these seek to solve the primal and dual barrier subproblems (4), (6) simultaneously. The main equation to be solved has the same form (5) as for the primal barrier method, except that the diagonal matrix X^2 is replaced by XZ^{-1}.

The equations $Ax = b$, $A^T\pi + z = c$, $x \geq 0$, $z \geq 0$ are satisfied throughout, and the iterations work towards satisfying the complementarity condition $XZ = \mu I$ as $\mu \rightarrow 0$.

5.1. Results

At first sight, the primal-dual approach would seem to incur the disadvantages of both primal and dual algorithms, at least in terms of obtaining initial interior points.

Nevertheless, McShane *et al.* [28] have given computational results for a primal-dual implementation, as well as for the dual affine code of [32] (evidently further refined). On the 30-problem subset mentioned above, both codes showed an average speed-up of about 4 relative to MINOS 5.0.

6. A Single-Phase Dual Barrier Method

In order to run *all* of the test problems in the *netlib* collection it is necessary to develop an algorithm that treats the primal LP problem in its most general form:

$$\text{Primal:} \quad \underset{x}{\text{minimize}} \quad c^Tx \quad \text{subject to} \quad Ax = b, \quad l \leq x \leq u. \tag{7}$$

The corresponding dual problem may be written

$$\begin{aligned} \text{Dual:} \quad & \underset{\pi, y, z}{\text{maximize}} && b^T\pi - u^Ty + l^Tz \\ & \text{subject to} && A^T\pi - y + z = c, \qquad y,\ z \geq 0. \end{aligned} \tag{8}$$

An unexpected advantage arising from this formulation is that the dual constraints $A^T\pi - y + z = c$ are easily satisfied by a suitable choice of the slacks y and z.

Further details are given in Gill *et al.* [17].

6.1. Results

An implementation to solve (8) is currently under development at Stanford. In preliminary tests, speed-up factors in the range 1 to 5 relative to MINOS 5.2 have been obtained for most of the first 53 problems in *netlib*.

These results are similar to those obtained for the primal barrier implementation [16] mentioned in Section 3.3, which is designed for problem (7). Both codes use essentially exact Cholesky factors of AX^2A^T, with iterative refinement and a partitioning (Schur-complement) scheme to handle dense columns of A. It is hoped that the dual implementation will show an advantage with inexact factorization and preconditioning (Section 7.3).

7. Computational Matters

The issues of starting points, rank-deficiency, termination criteria, restarts, etc. are too lengthy to discuss here. We choose just four topics.

7.1. Normal equations versus least squares

In *all* of the interior-point algorithms, the search direction is obtained from a system of the form

$$AX^2A^Tq = v, \tag{9}$$

where X is a diagonal matrix and v depends on the algorithm. If the right-hand side happens to be of the form $v = AX^2r$, this system is a set of "normal equations" equivalent to the linear least-squares problem

$$\min \|X(r - A^Tq)\|_2. \tag{10}$$

Least-squares prob' ms can be solved more reliably if treated as such. For example, conjugate-gradient methods generally require more iterations to solve (9) than they do to solve (10), particularly when the matrix XA^T is ill-conditioned (as it invariably is in this context).

We note that *not all interior-point methods permit the least-squares formulation.* In this respect, the advantage lies with the primal and primal-dual variants, and with the single-phase dual algorithm.

To date, all implementations except [18,19] have used the (less reliable) normal-equations approach.

7.2. Boundedness of the projections

A clue to the survival of interior-point implementations in the face of extreme ill-conditioning of XA^T has recently been provided by Stewart [39]. In the notation of equation (10), Stewart shows that as long as X is diagonal with nonzero elements and A has full row rank, the weighted pseudo-inverse $A^{\dagger}_X = (AX^2A^T)^{-1}AX^2$ and the oblique projection $A^TA^{\dagger}_X$ are both bounded in norm, independently of X.

Briefly, this means that q and A^Tq in (10) will not be arbitrarily large, even if X causes AX^2A^T to be almost singular.

In practice, of course, Cholesky factorization may fail if AX^2A^T is almost singular, and other factorizations may be needed; see Section 7.4.

7.3. Preconditioning

In the linear programming context, virtually all implementors have used *exact* Cholesky factors of AX^2A^T (excluding perhaps a few dense columns of A). The main reason is that the sparsity pattern of the normal-equations matrix does not change as X changes; hence a single "analyze" can be performed on the sparsity pattern of AA^T to obtain an ordering of the rows of A that preserves the sparsity of the Cholesky factor. The same ordering is used for all subsequent factorizations $AX^2A^T = R^TR$.

An exception is Karmarkar himself. In talks such as [24] he advocates forming the normal-equations matrix only approximately:

$$M = AX^2A^T + E_1,$$

and then factoring M approximately:

$$N(M+\sigma I)N^T = I + E_2,$$

where σ is an undocumented spectral shift, and E_1 and E_2 are "small" matrices representing the error involved. The matrix N is then used as a preconditioner when the conjugate-gradient method is applied to (9). (It could also be used with the least-squares formulation (10).)

A further device is to retain the *same* preconditioner N for several iterations of the main algorithm (perhaps modifying N slightly but cheaply at each iteration [23,38]).

Are these the keys to Karmarkar's spectacular run-times? Perhaps so, at least for some problems. Preconditioning of this kind is widely used in other areas and has been studied in the LP context by Gay [11]. Further study is well justified, as it is for LU preconditioning (Section 3.5). We note that as convergence occurs, the preconditioner must become more exact. For general problems, this can be achieved easily with LU factors, but *not* necessarily with Cholesky factors.

7.4. Alternative factorizations

While Cholesky factors of AX^2A^T have been preferred to date on efficiency grounds, we mention that greater reliability is likely with orthogonal and symmetric-indefinite factorizations of the form

$$XA^T = Q\begin{pmatrix} R \\ 0 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} = U^TDU.$$

For the case where A is sparse, considerable progress has been made recently by George, Liu and Ng [15] and Duff [7]. Dense columns in A remain a complication for some problems. Here the dual algorithms have an advantage in not requiring a (dense) artificial column for Phase 1.

8. Promising Approaches

Before closing, we review some methods that have yet to be proven computationally but appear to have significant computational advantages.

8.1. Eriksson's algorithm

The primal-dual approach was discussed in Section 5. A significantly *different* primal-dual algorithm was given in 1981 by Eriksson [8] and further developed in [9]. A primal subproblem is used, based on an *entropy function* and a current estimate of x (x^k):

$$\begin{array}{lll} \text{Primal:} & \underset{x}{\text{minimize}} & c^Tx - \mu\sum_{j=1}^{n}\left\{x_j\ln(x_j/x_j^k) - (x_j - x_j^k)\right\} \\ & \text{subject to} & Ax = b. \end{array} \tag{11}$$

It is assumed that $x^k > 0$, but not that $Ax^k = b$. Instead, the dual of (11) is treated as an unconstrained problem in π:

$$\text{Dual:} \qquad \underset{\pi}{\text{maximize}} \quad b^T\pi - \mu \sum_{j=1}^{n} x_j^k e^{-z_j/\mu}, \tag{12}$$

where $z = c - A^T\pi$. An inexact Newton method is applied to (12), with the central system of equations taking the form

$$AXA^Tq = b - Ax^k. \tag{13}$$

This algorithm has many intriguing properties, and we believe it to be of great promise. For example, the matrix AXA^T will in general be better-conditioned than the usual AX^2A^T. Competitive computational results await implementation of a sparse preconditioner for (13), using techniques that have been applied to barrier algorithms elsewhere.

8.2. The Box Method

One interpretation of the affine invariant algorithms is that they optimize the objective function subject to feasibility and a quadratic constraint. The ellipsoid defined by the quadratic constraint may be replaced by a "box".

Such an approach has been described by Zikan and Cottle [41,42], and promising computational results have been obtained for the special case of network problems. For more general linear programs, an appealing feature is that a normal simplex-type basis factorization $B = LU$ is suitable for the main steps of the algorithm. In addition, large numbers of columns of B can be replaced in the basis at once.

8.3. Shifted barrier methods

Apart from Eriksson's algorithm, the only method that specifically allows an arbitrary starting point is the shifted barrier approach (Gill *et al.* [21]). For problem (2) the objective function is of the form

$$F_{w,s}(x) = c^Tx - \sum w_j \ln(x_j + s_j),$$

where w is a set of weights corresponding to μ, and s is a set of shifts.

Apart from the ability to use a "good" starting point, an advantage is that the condition of the system used to obtain a search direction can be controlled by judicious choice of w and s. Convergence results have been established and a preliminary implementation has been developed at Stanford.

9. Conclusions

Some broad thoughts have already been given in Section 1.4. Many additional references could have been cited, all contributing to the current outlook.

In summary, we believe that the simplex method will remain the workhorse for the majority of existing applications, given practitioners' normal mode of operation: frequent restarts on slightly modified models.

However, from the last few years of experience we would say that interior-point algorithms are efficient on a rather larger class of (general) linear programs than we once would have supposed. New and *extremely large* applications will require decomposition if simplex techniques are to succeed. The alternative is a radically new approach, such as we seem to have here. While much trial and error lies ahead, we feel that the future for the new approach to linear programming shines challenging and bright.

In the meantime, we have a new way of identifying examples where the simplex method has performed poorly. We also have a renewed interest in improving the simplex method (e.g., [20,25]).

References

[1] I. Adler, N. Karmarkar, M. G. C. Resende and G. Veiga (1987). Data structures and programming techniques for the implementation of Karmarkar's algorithm, Manuscript (December 1987), Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA.

[2] I. Adler, M. G. C. Resende and G. Veiga (1986). An implementation of Karmarkar's algorithm for linear programming, Report ORC 86-8, Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA.

[3] S. Chen (1986). Computational experience with the Karmarkar algorithm, presented at the TIMS/ORSA Joint National Meeting, Los Angeles, CA, April 1986.

[4] G. B. Dantzig (1963). *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ.

[5] R. S. Dembo, S. C. Eisenstat and T. Steihaug (1982). Inexact Newton methods, *SIAM Journal on Numerical Analysis* 19, 400–408.

[6] I. I. Dikin (1967). Iterative solution of problems of linear and quadratic programming, *Soviet Math. Doklady* 8, 674–675.

[7] I. S. Duff (1988). The solution of large-scale least-squares problems on supercomputers, presented at workshop on *Supercomputers and Large-scale Optimization*, University of Minnesota Supercomputer Institute, Minneapolis, MN.

[8] J. R. Eriksson (1981). Algorithms for entropy and mathematical programming, Ph.D. thesis, Department of Mathematics, Linköping University, Linköping, Sweden.

[9] J. R. Eriksson (1985). An iterative primal-dual algorithm for linear programming, Report LiTH-MAT-R-1985-10, Department of Mathematics, Linköping University, Linköping, Sweden.

[10] D. M. Gay (1985a). Electronic mail distribution of linear programming test problems, *Mathematical Programming Society COAL Newsletter* 13, 10–12.

[11] D. M. Gay (1985b). Solving sparse least-squares problems, Seminar, Department of Operations Research, Stanford University, Stanford, CA.

[12] D. M. Gay (1987). A variant of Karmarkar's linear programming algorithm for problems in standard form, *Mathematical Programming* 37, 81–90.

[13] D. M. Gay (1988). Massive memory buys little speed for complete, in-core sparse Cholesky factorization, Numerical Analysis Manuscript 88-04, AT&T Bell Laboratories, Murray Hill, NJ.

[14] J. A. George and J. W. H. Liu (1981). *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, NJ.

[15] J. A. George, J. W. H. Liu and E. Ng (1988). A data structure for sparse QR and LU factorizations, *SIAM J. on Scientific and Statistical Computing* 9, 100–121.

[16] P. E. Gill, A. Marxen, W. Murray, M. A. Saunders and M. H. Wright (1988). Newton barrier methods for linear programming: some implementation aspects, presented at TIMS/ORSA Joint National Meeting, Washington, DC.

[17] P. E. Gill, W. Murray and M. A. Saunders (1988). A single-phase dual barrier method for linear programming, Report SOL 88-10, Department of Operations Research, Stanford University, Stanford, CA.

[18] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin and M. H. Wright (1986). On projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method, *Mathematical Programming* 36, 183–209.

[19] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright (1986). A note on nonlinear approaches to linear programming, Report SOL 86-7, Department of Operations Research, Stanford University, Stanford, CA.

[20] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright (1988a). A practical anti-cycling procedure for linear and nonlinear programming, Report SOL 88-4, Department of Operations Research, Stanford University, Stanford, CA.

[21] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright (1988b). Shifted barrier methods for linear programming, Report SOL 88-9, Department of Operations Research, Stanford University, Stanford, CA.

[22] C. C. Gonzaga (1987). Search directions for interior linear programming methods, Memorandum UCB/ERL M87/44, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA.

[23] N. Karmarkar (1984). A new polynomial-time algorithm for linear programming, *Combinatorica* 4, 373–395.

[24] N. Karmarkar (1987). Recent developments in new approaches to linear programming, presented at the SIAM Conference on Optimization, Houston, TX, May 1987.

[25] E. S. Klotz (1988). Dynamic pricing criteria in linear programming, Ph.D. dissertation, Department of Operations Research, Stanford University, Stanford, CA.

[26] I. J. Lustig (1987). An analysis of an available set of linear programming test problems, Report SOL 87-11, Department of Operations Research, Stanford University, Stanford, CA.

[27] I. J. Lustig (1988). A generic primal-dual interior point algorithm, Report SOR 88-3, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[28] K. A. McShane, C. L. Monma and D. Shanno (1988). An implementation of a primal-dual interior point method for linear programming, Manuscript (March 1988), RUTCOR, Rutgers University, New Brunswick, NJ.

[29] N. Megiddo (1986). Pathways to the optimal set in linear programming, Report RJ 5295, IBM Almaden Research Center, San Jose, CA.

[30] N. Megiddo and M. Shub (1986). Boundary behaviour of interior point algorithms in linear programming, Report RJ 5319, IBM Thomas J. Watson Research Center, Yorktown Heights, NY.

[31] C. L. Monma (1987). Recent breakthroughs in linear programming methods, Manuscript, Bell Communications Research, Morristown, NJ.

[32] C. L. Monma and A. J. Morton (1987). Computational experience with a dual affine variant of Karmarkar's method for linear programming, *Operations Research Letters* 6, 261–267.

[33] R. D. C. Monteiro and I. Adler (1987a). Interior path following primal-dual algorithms—Part I: Linear programming, Technical Report, Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA.

[34] R. D. C. Monteiro and I. Adler (1987b). An $O(n^3L)$ primal-dual interior point algorithm for linear programming, Technical Report, Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA.

[35] R. D. C. Monteiro, I. Adler and M. G. C. Resende (1988). A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series expansion, Report ESRC 88-8, Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA.

[36] B. A. Murtagh and M. A. Saunders (1983). MINOS 5.0 user's guide, Report SOL 83-20, Department of Operations Research, Stanford University, Stanford, CA.

[37] J. Renegar (1988). A polynomial-time algorithm, based on Newton's method, for linear programming, *Mathematical Programming* 40, 59–93.

[38] D. F. Shanno (1988). Computing Karmarkar projections quickly, *Mathematical Programming* 41, 61–71.

[39] G. W. Stewart (1988). On scaled projections and pseudo-inverses, Report TR-??, Department of Computer Science, University of Maryland, College Park, MD.

[40] R. J. Vanderbei, M. S. Meketon and B. A. Freedman (1986). A modification of Karmarkar's linear programming algorithm, *Algorithmica* 1, 395–407.

[41] K. Zikan and R. W. Cottle (1987a). The Box Method for linear programming: Part I—Basic theory, Report SOL 87-6, Department of Operations Research, Stanford University, Stanford, CA.

[42] K. Zikan and R. W. Cottle (1987b). The Box Method for linear programming: Part II—Treatment of problems in standard form with explicitly bounded variables, Report SOL 87-9, Department of Operations Research, Stanford University, Stanford, CA.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
Technical Report SOL 88-14

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Interior-Point Methods for Linear Programming:
A Challenge to the Simplex Method

5. TYPE OF REPORT & PERIOD COVERED
Technical Report

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Philip E. Gill, Walter Murray and
Michael A. Saunders

8. CONTRACT OR GRANT NUMBER(s)
#N00014-87-K-0142
#AFOSR-87-0196

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Department of Operations Research - SOL
Stanford University
Stanford, CA 94305-4022

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
1111MA

11. CONTROLLING OFFICE NAME AND ADDRESS
Office of Naval Research - Dept. of the Navy
800 N. Quincy Street
Arlington, VA 22217

12. REPORT DATE
July 1988

13. NUMBER OF PAGES
14 Pages

Air Force Office of Scientific Research/NM
Building 410
Bolling Air Force Base
Washington, DC 20332

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Linear Programming, Interior-Point Methods, Barrier-Function Methods, Newton's Method, Karmarkar's Projective Method.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The world of mathematical programming has seen a remarkable surge of activity following publication of Karmarkar's projective algorithm in May 1984. A review of the ensuing three years has already appeared (Monma [31]). One year later, we review some of the main methods and surrounding events, and focus on references that contain computational results.

END

DATE

FILMED

DTIC

11-88